Amendments to the Specification:

Applicants request that paragraphs 0005, 0007, 0043, 0052, 0053, 0056, 0058, 0069, 0073, and 00100 be replaced in the specification with the following paragraphs respectively:

[0005]   An application server 102 is often used to host a variety of applications (such as application 103). Business logic application software and/or database application software are frequent types of application software that are hosted by an application server ~~101~~ 102. Here, "hosting" generally means being responsible for interpreting and/or formatting messages received/sent to network 101 so that the application 103 is properly used by the enterprise. For example, in a basic case where application ~~101~~ 103 is a business logic application, the application server 102 responds to a request from the network 101 for application 103 (i.e., a request from some entity that has expressed a need for application 103 through network 101) by properly invoking application 103 in response to the request; and, forwards the result(s) of the application's execution to the requestor.

[0007]   Functional elements 104 – 108 depict a web server 104 and its corresponding Java based "back-end" functionality 105 – 108. The term "web server" 104 is largely understood to mean being capable of presenting a web based interface (e.g., through the downloading of web pages scripted in HTML (HyperText Markup Language) format) over a network 101. Accesses to specific web pages associated with the web based presentation are typically formatted in the HTTP (HyperText Transfer Protocol) protocol. Often, useful tasks that are dependent on

business logic and/or database functions are made accessible through a web based presentation. **Figure 1** suggests such an approach by way of the back end <u>JSP (Java Server Page)</u> servlet engine 105, database (DB) 106 and Enterprise Java Beans (EJB) 107 applications, and J2EE <u>(Java Platform, Enterprise Edition)</u> server 108.

[0043] Embodiments between the extremes discussed above are also inherently supported. Irrespective of how many GRM applications exist per scenario, GRM application 210 may be implemented as a servlet (having its own unique URL <u>(Uniform Resource Locator)</u>) that is dedicated to execute the software component availability testing for its constituent scenario(s). Request message 211 would therefore identify the URL of GRMG application 210 so that it could be executed as a consequence.

[0052] If so, the GRMG infrastructure is expected to refer to the appropriate region of the customizing file in order to generate a request message for a specific scenario (e.g., a first section of the customizing file is referred to in order to generate a request message for a first scenario; a second section of the customizing file is referred to in order to generate a request message for a second scenario;, etc.). **Figure 4** shows an embodiment of an organization scheme for a customizing file that includes information for each of a plurality of scenarios (so that a unique request message can be generated for each scenario). The organization scheme entails listing basic control information 401 as well as the information for each the scenarios 402_1 through 402_X. In an embodiment, the customizing file is in the format of a document that is capable of supporting the execution of software (e.g., an .XML

(Extensible Markup Language) document). As such, the information is embodied in the appropriate format for the document.

[0053]   According to the embodiment of **Figure 4**, the basic control information 401 is used to control the execution of the GRMG infrastructure itself and includes a "run" field 401_1; a "runlog" field 401_2; and, a "runerror" field 401_3. The run field 401_1 specifies whether the GRMG infrastructure that would use the customizing file is running or not. In a further embodiment, the customizing file is in the form of a document such as an .XML ~~,XML~~ document. ~~Here~~ As an example, an "X" ~~is~~ marked at ~~an~~ the appropriate location in the .XML document may ~~to~~ indicate that ~~whether or not~~ the applicable GRMG infrastructure is running and the lack of an "X" may indicate that the corresponding GRMG infrastructure is not running ~~(e.g., X = running; no X = not running)~~.

[0056]   Items 403 through 410 correspond to "control" items that apply to the scenario as a whole while items 411_1 through 411_x correspond to bodies of information that pertain to a specific component that is to be tested for availability. The scenario name field 403 provides the name of the scenario. The scenario version field 404 provides the version of the scenario.

[0058]   The scenario type field 406 identifies how the appropriate GRMG application for the scenario instance is to be reached. Here, as there exist a number of different ways in which executable routines may be called upon, the manner that is identified in the scenario type field 406 should be consistent with the manner in which the GRMG application that is to be executed for the scenario instance has been implemented. For example, if the appropriate GRMG application for the

scenario instance is a Java servlet (e.g., to be executed by servlet engine 205 of **Figure 2**) that is reachable with a URL address; then the scenario type field 406 will indicate that a URL is to be specified in the request message. Alternatively, if the appropriate GRMG application for the scenario instance is reachable with an RFC (Remote Function Call) destination, the scenario type field 406 will indicate that HTTP should be used in the sending of the request message.

[0069] Here, the request message 502 could be crafted 502 simply by copying the content of: 1) the scenario name field 403 of the customizing file into the scenario name field 503 of the request message; 2) the scenario version field 404 of the customizing file into the scenario version field 504 of the request message; 3) the scenario instance field 405 of the customizing file into the scenario instance field 505 of the request message; and, 5) attributes of the component specific information fields 411_1 – 411_X of the customizing file into the component specific information fields 511_1 – 511_X of the request message. For example, as applied to the "X^{th}" component, the content of the component name, component version, property name and property value fields of the customizing file 412, 413, 416, 418 could be copied into the component name, component version, property name and property value fields of the request message 512, 513, 516, 518.

[0073] Embodiments between the extremes discussed above are also inherently supported. Irrespective of how many GRMG applications exist per scenario, GRMG application 210 may be implemented as a servlet (having its own unique URL) that is dedicated to execute the software component availability testing for its constituent

scenario(s). Request message 211 would therefore identify the URL of GRMG application 210 so that it could be executed as a consequence.

[00100] Actual messages pertaining to the component that were contained in received response messages can be displayed next to the heartbeat node 806. Recalling the discussion of the response messages that was provided above, and referred to **Figure 7**, recall that items for conveying messages 722 were specifically provided for. Here, for example, the actual textual message (e.g., "component is alive") to which a numeric encoding was provided by way of the area, number (and perhaps number) fields $722_3 - 722_5$ in a received response message for the component may be displayed next to the heartbeat node 806 in the display. Likewise, a textual message received in a response message for a component (e.g., as contained in field 722_6 of the response message embodiment of **Figure 7**) can also be displayed. In an alternative embodiment, a status indication is provided rather than message text.